METHOD AND APPARATUS FOR INITIALIZING MULTIPLE SECURITY MODULES

BY:

MICHAEL F. ANGELO
3303 AMBER FOREST DRIVE
HOUSTON, TX 77068

LARRY N. MCMAHAN
41237 CHILTERN DRIVE
FREMONT, CA 94539

RICHARD D. POWERS
2985 HILLSIDE DRIVE
HIGHLAND VILLAGE, TX 75077

METHOD AND APPARATUS FOR INITIALIZING MULTIPLE SECURITY MODULES

BACKGROUND OF THE RELATED ART

[0001] This section is intended to introduce the reader to various aspects of art, which may be related to various aspects of the present invention that are described and/or claimed below. This discussion is believed to be helpful in providing the reader with background information to facilitate a better understanding of the various aspects of the present invention. Accordingly, it should be understood that these statements are to be read in this light, and not as admissions of prior art.

[0002] In the field of processor-based systems, such as computer systems, it may be desirable for information to be transferred from one system to another system via a network. Networks may be arranged to allow information, such as files or programs, to be shared across an office, a building, or any geographic boundary. While these networks may be used to increase productivity, they also expose computer systems to security risks, such as interception of confidential data by unauthorized parties, loss of data integrity, unauthorized access to computer systems on the network, and the like.

[0003] A wide variety of security measures may be employed to secure data in a networked environment. For example, security components or modules may be used to attest to the settings within the computer system. In other words, the security modules may certify the validity of the computer system as a system that may be trusted by other systems. Such a security module may maintain records that relate to components and devices that are utilized by the computer system. These records, which may be encrypted with keys to prevent

unauthorized access and/or may be managed by an administrator who is able to control the configuration of the computer system. However, if multiple security modules are utilized in a single computer system, each security module may contain incompatible data, which may be undecipherable by the computer system or other security modules. The data is incompatible because each security module may have a different perspective (i.e. may store differently encrypted data) of the information within the computer system and/or the operating environment of the computer system. These differences in perspective may result in conflicts that prohibit effective operation of the computer system.

[0004] For example, if two security modules are utilized in a computer system, each of the security modules may have different information that relates to the configuration of the computer system because each security module may have its own keys to encrypt the information that relates to the computer system. If one of the security modules is damaged, access to the information that was stored in or for the security module may not be obtainable. As such, the loss of a single security module may hinder the operation of the computer system as a whole and prevent access to specific information within the computer system.

[0005] In addition, with multiple security modules in a single computer system, the security modules may not be able to determine which security module is the controlling security module. As a result, the security modules may be unable to coordinate the operation of multiple security modules within a single computer system. Also, an administrator of the computer system may not be able to determine which security module is being utilized by the computer system. This problem may hinder maintenance or troubleshooting of the computer system.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] Exemplary embodiments of the present invention may be apparent upon reading of the following detailed description with reference to the drawings in which:

[0007] FIG. 1 is a block diagram illustrating a network in accordance with embodiments of the present invention;

[0008] FIG. 2 is a block diagram illustrating an exemplary security module in accordance with embodiments of the present invention;

[0009] FIG. 3 is a block diagram illustrating a computer system with multiple security modules in a network in accordance with embodiments of the present invention;

[0010] FIG. 4 is a process flow diagram illustrating the initialization of a security module in accordance with embodiments of the present invention;

[0011] FIG. 5 is a process flow diagram illustrating the use of a lock bit to coordinate the initialization of multiple trusted platform modules in accordance with embodiments of the present invention;

[0012] FIG. 6 is an alternative process flow diagram illustrating the use of a lock bit to coordinate the initialization of multiple trusted platform modules in accordance with embodiments of the present invention; and

[0013] FIG. 7 is a process flow diagram illustrating the use of a lock bit and a genkey fail bit to coordinate the initialization of an additional trusted platform module in accordance with embodiments of the present invention.

DESCRIPTION OF SPECIFIC EMBODIMENTS

[0014] One or more specific embodiments of the present invention will be described below. In an effort to provide a concise description of these embodiments, not all features of an actual implementation are described in the specification. It should be appreciated that in the development of any such actual implementation, as in any engineering or design project, numerous implementation-specific decisions may be made to achieve the developers' specific goals, such as compliance with system-related and business-related constraints, which may vary from one implementation to another. Moreover, it should be appreciated that such a development effort might be complex and time consuming, but would nevertheless be a routine undertaking of design, fabrication, and manufacture for those of ordinary skill having the benefit of this disclosure.

[0015] The Trusted Computing Platform Alliance, which includes the assignee of the present application, is developing specifications that are intended to improve security for computing systems. Two such specifications under development are the Trusted Computing Platform Alliance Trusted Platform Module Protection Profile Specification and the Trusted Computing Group ("TCG") Main Specification, which are hereby incorporated by reference. These specifications refer to a trusted platform module ("TPM"), which is defined as a module that includes protected functionality and shielded locations.

[0016] Embodiments of the present invention may provide a methodology for initializing multiple security modules, such as TPMs, in a computer system. A lock bit may be employed in a multiple TPM environment to determine the TPM that is the controlling TPM. The lock bit may be located in non-volatile memory, and be accessible by various TPMs within the computer system. By utilizing the lock bit, one TPM may generate keys and measure the computer system. The keys and measured system information may be copied into the other TPMs. As a result, each of the TPMs may share a common view of the computer system and provide fault tolerance in the event of the failure of one of the TPMs. In other words, the TPMs may each have consistent information and be able to function compatibly with each other. Thus, the computer system may implement multiple TPMs to provide redundancy and enhance the operation of the computer system.

[0017] Referring initially to FIG. 1, a block diagram of a computer network architecture is illustrated and designated using a reference numeral 10. A server 20 may be connected to a plurality of client computers 22, 24 and 26. The server 20 may be connected to as many as "n" different client computers. The magnitude of "n" may be a function of the computing power of the server 20. Each client computer in the network 10 may be a functional client computer and may be a desktop personal computer ("PC"), a notebook PC, a tablet PC, a personal digital assistant ("PDA"), or the like.

[0018] The server 20 may be connected via a network infrastructure 30, which may include a combination of hubs, switches, routers, or the like. While the network infrastructure 30 is illustrated as being either a local area network ("LAN"), a wide area network ("WAN"), or a metropolitan area network ("MAN"), those skilled in the art will

appreciate that the network infrastructure 30 may assume other forms or may even provide network connectivity through the Internet. As described below, the network 10 may include other servers as well, which may be dispersed geographically with respect to each other to support client computers in other locations.

[0019] The network infrastructure 30 may connect the server 20 to a server 40, which may be representative of any other server in the network environment. The server 40 may be connected to one or more client computers 42, 44, and 46. As illustrated in FIG. 1, a network infrastructure 50, which may include a LAN, a WAN, a MAN, or other network configuration, may be used to connect the client computers 42, 44 and 46 to the server 40. The server 40 may additionally be connected to the Internet 60, which may be connected to a server 70. The server 70 also may be connected to one or more client computers 72, 74 and 76.

[0020] In the network infrastructures 30, 50 and 60, the systems, such as the client computers 22-26, 42-46 and 72-76 and servers 20, 40, 70, may be subject to improper access attempts, such as hacker attacks, disruption of service attacks, introduction of malicious code, viruses and the like. These attacks may result in a loss of productivity, revenue, data, and/or confidential information that is stored on one of the systems. To protect the data and the system, security modules, such as TPMs, may be utilized to provide enhanced security. An exemplary security module, which provides this functionality, is illustrated in FIG. 2.

[0021] FIG. 2 is a block diagram of an exemplary security module in accordance with the present invention. The security module 80 may include various components, such as a detector 82, a key generator 84, and a key receiver 86 along with other components (not

shown). The detector 82, the key generator 84, and the key receiver 86 may be hardware, software, or any combination thereof, which may be individual components or combined into a single device. The detector 82 may be utilized to determine if the security module 80 is a controlling security module or a subordinate security module, which is discussed below in greater detail. The key generator 84 may generate a key or keys for the security module if the security module is the controlling security module. The key generator 84 may be utilized to encrypt information to be stored on the system or even within the security module 80. Also, a key receiver 86 may receive a key or keys from the controlling security modules if the security module is the subordinate security module. The use and interaction of these various components is further explained below.

[0022] These various components 82-86 may be utilized by the security module 80 to enhance the security of the system. For example, each of the client computer systems and servers, which are described in FIG. 1, may include a TPM to provide integrity for that system on the network. However, with a single TPM, the user may lose access to the system if the TPM fails. As such, to provide fault tolerance, multiple TPMs may be implemented within a system to enhance the security and reliability of the system.

[0023] FIG. 3 is a block diagram illustrating an exemplary computer system with multiple security modules in accordance with embodiments of the present invention. The computer system is generally referred to by the reference numeral 100. The architecture of the computer system 100 is given for purposes of illustration only, as one example of a computer system in which embodiments of the present invention may be employed. Additionally, it should be appreciated that any number of security modules, such as security modules 80 or TPMs, may be utilized by the system 100, and connected in a variety of

locations within the computer system 100. Two security modules are depicted in the system illustration in FIG. 3.

[0024] The computer system 100 may comprise a processor complex 102, which may include one or more central processing units ("CPUs"). A core logic chipset 104, which may manage a variety of functions on behalf of the processor complex 102, may be connected to the processor complex via a processor interface point-to-point link or a processor bus 103.

[0025] The core logic chipset 104 may be connected via memory bus 105 to a system random access memory 106, which may comprise static random access memory ("SRAM"), dynamic random access memory ("DRAM") or other suitable memories. The memory 106 may be a shared system memory to hold memory resident files or information. A video graphics controller 110 may be connected to the core logic chipset 104 via a video bus 111 to provide a signal that produces a display image on a video display 112.

[0026] A bus 113, such as a peripheral component interface ("PCI") bus or the like, may connect the core logic chipset 104 to a variety of system devices, such as a network interface card 122 and a PCI/PCI bridge 124. The network interface card 122 may provide communication capability to the computer system 100 via a communication bus 119. The communication bus 119 may be connected to other computer systems, as discussed above. The PCI/PCI bridge 124 may provide capacity for additional PCI devices on a PCI bus 117.

[0027] A PCI/SCSI bus adapter 114 may provide access to SCSI devices such as a disk drive 130 and a tape drive 132 via a SCSI bus 131. A PCI/ATA controller 118 may provide access to additional devices, such as a disk drive 128 and a CD ROM drive 134. A

PCI/EISA/LPC bridge 116 may provide access to system devices, such as a read only memory basic input/output system ("ROM BIOS") 139, a non-volatile memory ("NVRAM") 140, a modem 120, a first trusted platform module ("TPM") 143 or the like via a bus 138. The operation of the first TPM 143 is discussed below in greater detail. Also, the NVRAM 140 may include flash memory or the like and may include a lock bit 141 and/or a genkey fail bit 142. The operation of the lock bit 141 and the genkey fail bit 142 are also discussed below. The BIOS 139 may also be system firmware that is stored in ROM. The BIOS 139 may be referred to as the core root of trust for measurement ("CRTM"), which is the basis for insuring the integrity of the computer system 100. As such, the BIOS 139 provides the foundation for trust, which makes and reports trust measurements of components external to the first TPM 143. The measurements may include cryptographically hashing code or configurations to create integrity metrics for the TPM. The modem 120 may provide communication access via a phone line 121. An input/output controller 126, which may be connected to the bus 138, may provide access to system devices such as a CD ROM drive 144, a keyboard 146, a mouse 148, a floppy disk drive 150, a serial port 152, a second TPM 153, a real time clock (" RTC") 154, and the like, via a bus 155.

[0028] The TPMs 143 and 153 may provide the computer system 100 with enhanced security because they may be used to validate the BIOS or system firmware along with other code. The TPMs 143 and 153 may include an input/output interface, a processor, and memory, routines, for example. These various components may be utilized to perform the functionality of the detector 82, the key generator 84, and the key receiver 86, which are discussed above in FIG. 2. The input/output interface may be utilized by the TPMs 143 and 153 to communicate with other components within the computer system 100 or to receive power. The processor in the TPMs 143 and 153 may be utilized to provide cryptographic

capabilities, such as hashing, random number generation, key generation, and encryption/decryption. The memory may be divided into registers and other memory sections, which may be utilized to store the keys and hashed information relating to code or configurations of the computer system 100. Because each of the TPMs 143 and 153 operate with the computer system 100, the TPMs 143 and 153 may use routines to attest to the integrity of the computer system 100. In other words, the TPMs 143 and 153 may certify that the computer system 100 is a valid system that may be trusted by other systems.

[0029] To provide enhanced security and establish root trust for the computer system, various security measures may be performed by the TPMs 143 and 153. For instance, each TPM may include endorsement keys, which are a private and public key pair that are used to encrypt/decrypt information. The endorsement keys may be unique to a particular TPM, and may be assigned to the TPM when it is manufactured. Also, an attestation identity key may be used to provide platform authentication along with a user key that may be used to provide privacy to a user of the TPMs 143 and 153. In addition to the keys, the TPMs 143 and 153 may include hashing capabilities and a random number generator to further enhance the security of the computer system by hashing information, such as code or configuration information about one or more system components.

[0030] The TPMs 143 and 153 may follow an initialization when they are first activated within the computer system 100. This initialization, which shall be referred to as TPM initialization, is different from the initialization that is performed on all devices in a computer system when the computer system is booted. That initialization shall be referred to as system initialization. During the TPM initialization, which is a routine performed by the TPM, system state information and keys may be stored in the memory of the TPM. The TPM

initialization process may include the validation of the BIOS 139 by the TPM 143 or 153 to establish trust with the BIOS 139, and the validation of other code and configurations by the BIOS 139 to build trust within the computer system 100. During the TPM initialization process, the ownership and identity of the TPM 143 or 153 in relation to the computer system 100 may be established. Ownership may be established by providing or generating keys for the TPM 143 or 153 and measuring the code and configuration of the computer system 100. The TPM initialization process is shown in greater detail in FIG. 4.

[0031] In FIG. 4, which is generally referred to by reference numeral 200, an exemplary initialization process of a TPM (i.e. TPM initialization), such as the TPMs 143 or 153 (FIG. 3), is shown. Each TPM in a given system may undergo the TPM initialization process illustrated in FIG. 4. The process begins at block 202. At block 204, the TPM may perform a self-test. The self-test may include verifying the operation of the internal components and information within the TPM. During the self-test, the TPM may generate keys, such as endorsement keys, for example. The keys may include private and public keys that may be used by the TPM to encrypt/decrypt different information. In addition, the self-test may include measuring various code or configurations, such as the BIOS 139 (FIG. 3), which may include other system firmware, and the BIOS boot block, if present. The measurement of a command or code may include cryptographically hashing the code to create integrity metrics. The hash may be a digital signature that provides authentication for the specific TPM through the use of private keys. At block 206, the TPM may store the measurements in its internal memory. The measurements may be stored in specific registers that are utilized by the TPM to store information relating to code or configurations, such as the BIOS and/or BIOS boot block, if present.

[0032] Once the BIOS has been measured by the TPM, the BIOS may be utilized to measure option read-only memory ("ROMs") and hardware, as shown in block 208. The option ROM may include programs associated with devices attached to system buses. The hardware may include various buses or devices within the computer system, as discussed above in FIG. 3. At block 210, the measurements from the BIOS of the option ROMs and hardware are stored within the TPM. Then, the BIOS may measure the option ROM configuration, other code and configurations, as shown in block 212. The other code and configurations may include the operating system ("OS") loader, the disk boot record, other code and data utilized to prepare the OS, state transitions, and/or wake events. After the measurements are made, the TPM may store the measurements in its internal memory, as shown in block 214. Next, the computer system boots, as shown in block 216. The booting of the system may include activating or handing control of the system to the operating system. The process ends at block 218.

[0033] If multiple TPMs are deployed within a computer system, each TPM may have different measurement information stored in its internal memory. This information may be referred to as the "perspective" of the TPM. The perspectives of the TPMs may differ because each TPM has different keys that are utilized to measure the system. If the TPMs within a system have different perspectives, system incompatibilities may result. For example, one of the TPMs cannot be used to back up the other TPM because the information within each TPM is unique.

[0034] Accordingly, for each of the TPMs to have the same perspective, the keys from one TPM may be copied into the other TPM. However, when the TPMs are initializing, it is unclear which TPM is the controlling TPM for the system. In other words, it may be unclear

which TPM is supposed to copy its keys to the other TPMs. As a result, inconsistencies may appear because each of the TPMs may attempt to copy keys into the other TPM. The inconsistencies may result in synchronicity issues that may degrade the performance of the system. In addition, an administrator of the system may not know which TPM is being utilized by the system. As a result, maintenance or troubleshooting of the system may be hindered. Accordingly, because it is unclear which TPM may be utilized to copy the keys into the other subordinate TPM, a mechanism for coordinating the TPMs may be utilized to prevent any inconsistencies or arbitrate between the TPMs within the system.

[0035] FIG. 5 is a process flow diagram illustrating the use of a lock bit to coordinate the initialization of multiple trusted platform modules in accordance with embodiments of the present invention. The process is generally referred to by reference numeral 300. To coordinate the initialization of the TPMs, each TPM within the system may access a lock bit, which may be a setting or attribute stored in memory, such as the NVRAM 140 (FIG. 3). One of the TPMs may set the lock bit to indicate that it is the controlling TPM for the computer system. The TPM that sets the lock bit may be the first TPM to access the lock bit, the TPM associated with the processor, or other factors may be used to determine the controlling TPM. Beneficially, by utilizing the lock bit, each TPM may determine if it is the controlling TPM or a subordinate TPM. After a controlling TPM is established, the keys from the controlling TPM may be copied into the other TPMs to allow the TPMs to have the same perspective of the system. With the same perspective, each of the TPMs may act as a back up for the other TPMs. Also, the inconsistencies encountered during the initialization process may be prevented because the lock bit is used to determine the TPM that provides the keys to the other TPMs.

[0036] The process begins at block 302. At block 304, the computer system may be activated by supplying power to multiple TPMs. The activation of the system may include providing power to various modules coupled to the system, such as the TPMs. Each TPM in the system may perform the process illustrated in FIG. 4. At block 306, the TPM may determine whether it has undergone the process of TPM initialization. If the TPM has previously undergone TPM initialization, it may utilize the information and keys within itself to measure the system, as shown in block 308. The measuring of the system may include the measurement of code and configurations, as previously discussed in blocks 204-214 (FIG. 3).

[0037] If a TPM has not previously undergone TPM initialization, then it may utilize the lock bit to determine whether it is the controlling TPM. At block 310, the TPM may determine if it can set the lock bit. The determination may include accessing and changing the lock bit in memory. The lock bit may include a setting or specific value that is located in memory, such as system RAM 106 or NVRAM 140 (FIG. 3). The lock bit may be accessible by the TPMs that are coupled to the system to allow each TPM to determine if it is the controlling or subordinate TPM. If the lock bit can be set by the TPM, then that TPM may be defined as the controlling TPM. As shown in block 312, the controlling TPM may then generate keys that may be used to hash or measure code in the system. The code may include the BIOS, the BIOS boot block, or other code and configurations, as discussed above. At block 314, the keys may be copied into the other (subordinate) TPMs within the system. Once the keys are copied into the other TPMs, the lock bit may be cleared, as shown in block 316. Then, the TPM may measure the system in block 308, as discussed above.

[0038] However, if the lock bit cannot be set by a TPM during initialization, then that TPM may be defined to be a subordinate TPM. The subordinate TPM may wait for the lock

bit to be cleared in block 318. Then, the subordinate TPM may test the lock bit, as shown in block 320. At block 322, the TPM may determine if the lock bit has been cleared. To clear the lock bit, the controlling TPM may alter the information or setting within the memory. If the lock bit indicates that it is still set, then the subordinate TPM may test the lock bit again, as shown in block 320. However, if the lock bit has been cleared, then the subordinate TPM may measure the computer system, as shown in block 308. Once the computer system is measured, the TPM may boot the computer system, as shown in block 324. The booting of the computer system may be similar to the discussion of block 216 above. Accordingly, the process ends at block 326.

[0039] FIG. 6 is an alternative process flow diagram illustrating the use of a lock bit to coordinate the initialization of multiple trusted platform modules in accordance with embodiments of the present invention. The process flow diagram is generally referred to by reference numeral 400. Each of the TPMs within a system, such as the computer system 100 (FIG. 3), may perform the TPM initialization process illustrated in FIG. 6. As discussed above, the lock bit may be utilized to determine which TPM may be the controlling TPM for the computer system. In addition to providing keys to the subordinate TPM, the controlling TPM may provide measured system data for the subordinate TPM. Beneficially, by providing keys and measurement information, the computer system may initialize in a more efficient manner by decreasing the amount of time for the TPMs to initialize.

[0040] The process begins at block 402. At block 404, the system may be activated in a manner similar to block 304. At block 406, the TPM may determine whether it has undergone TPM initialization. If the TPM has undergone TPM initialization, then the TPM

may measure the system, as shown in block 408. Then, the computer system may boot, as shown in block 410.

[0041] If a TPM determines that it has not undergone TPM initialization, it may utilize the lock bit to determine whether it is the controlling TPM. At block 412, the TPM may determine if it can set a lock bit. The determination of whether the TPM can set of the lock bit may be similar to the previous discussion regarding block 310. If the lock bit is set by the TPM, then the TPM may be defined to be the controlling TPM for the system. After the lock bit is set, the controlling TPM may generate keys, which may be used to hash or measure code in the system, as shown in block 414. At block 416, the controlling TPM may measure the code and configuration of the system, as discussed above with respect to block 408. The keys and the measurements may be copied from the controlling TPM to the subordinate or other TPM, as shown in block 418. Once the keys and measurements are copied into the other (subordinate) TPM, the lock bit may be cleared, as shown in block 420. Then, the computer system may boot, as shown in block 410.

[0042] If the TPM determines it cannot set the lock bit, then the TPM may be defined to be a subordinate TPM. At block 422, the subordinate TPM may wait for the lock bit to be cleared. The clearing of the lock bit may be similar to the description of block 318. Then, the subordinate TPM may test the lock bit, as shown in block 424. At block 426, the subordinate TPM may determine if the lock bit has been cleared, which may be similar to the discussion of block 322. If the subordinate TPM determines that the lock bit is still set, then the subordinate TPM may test the lock bit again, as shown in block 424. However, if the lock bit has been cleared, then the TPM may boot the system, as shown in block 410. The process ends at block 428.

[0043] FIG. 7 is a process flow diagram illustrating the use of a lock bit and a genkey fail bit to coordinate the initialization of an additional trusted platform module in accordance with embodiments of the present invention. The process flow diagram is generally referred to by reference numeral 500. In addition to the TPM initialization process for the system, the replacement or addition of a TPM, such as TPM 143 or 153 (FIG. 3), may result in one TPM having previously undergone TPM initialization, and the other TPM not having undergone TPM initialization. As shown in FIG. 7, a process flow diagram may utilize a lock bit and a genkey fail bit, such as the genkey fail bit 142 (FIG. 3), to coordinate the initialization of an additional TPM. When adding a TPM that has not undergone TPM initialization to the system, a TPM that has previously undergone TPM initialization may be the controlling TPM or the subordinate TPM. To coordinate the TPM initialization of the added TPM, an initialization setting, such as the genkey fail bit, may be used to prevent inconsistency or arbitrate the control between the new and initialized TPMs. Beneficially, the genkey fail bit may allow the non-initialized and initialized TPMs to be coordinated within the system.

[0044] The process begins at block 502. At block 504, the system is activated, which may be similar to block 304. At block 506, the TPM may determine whether it has previously undergone TPM initialization, as referenced in block 306. If the TPM has undergone TPM initialization, then the TPM may set the genkey fail bit, as shown at block 508. The genkey fail bit may be a setting or specific value that is located in memory, such as system RAM 106 or NVRAM 140 (FIG. 3). The genkey fail bit is accessible by the TPMs associated with the system. At block 510, the TPM may determine if it can set a lock bit. The determination whether the TPM can set of the lock bit may be similar to the discussion regarding block 310. If the lock bit is set by the TPM, then the TPM may be defined to be the

controlling TPM. Accordingly, the controlling TPM may copy keys from itself to the other subordinate TPMs, as shown in block 512. At block 514, the lock bit may be cleared after the keys are copied into the other TPM. However, if the TPM cannot set the lock bit, then it may clear the genkey fail bit, as shown in block 516. The clearing of the genkey fail bit may include changing the information or setting within the section of memory associated with the genkey fail bit. Then, the TPM may measure the system in block 518. The measuring of the system may include the measurement of code and configurations, as previously discussed in blocks 204-214.

[0045] If the TPM has not previously undergone TPM initialization, then it may be a new or an additional TPM that is being added to the system. In such a case, the lock bit may be used to determine the controlling TPM. At block 520, the TPM may determine if it can set a lock bit. The determination may be similar to the discussion in block 510. If the lock bit is set by the TPM, then the TPM may generate keys for itself, as shown in block 522. At block 524, the TPM may determine if the genkey fail bit is clear. The determination of whether the genkey fail bit is clear may include examining a specific section of memory to determine a value or setting. If the genkey fail bit is set, then the TPM may be defined as a subordinate TPM. The subordinate TPM may clear the lock bit in block 525. Then, the subordinate TPM may copy the keys from the other TPM, as shown in block 526. At block 534, the subordinate TPM may wait for the lock bit to be cleared, which will be discussed below. However, if the genkey fail bit is cleared, then the TPM may be defined to be the controlling TPM. The controlling TPM may store the keys generated into itself, as shown in block 528. At block 530, the controlling TPM may copy the keys into the other (subordinate) TPM. After the keys are copied into the other TPM, the lock bit may be cleared, as shown in block 532. Then, the TPM may measure the system in block 518, as discussed above.

[0046] If the lock bit cannot be set in block 520, then the TPM may be defined to be the subordinate TPM. The subordinate TPM may wait for the lock bit to be cleared in block 534. Then, the subordinate TPM may test the lock bit, as shown in block 536 and previously discussed in block 318. At block 538, the subordinate TPM may determine if the lock bit has cleared. If the lock bit has been cleared, then the TPM may measure the system in block 518, as discussed above. Once the system has been measured by the TPM, the system may boot, as shown in block 540 and discussed above in block 216. The process ends at block 542.

[0047] Alternatively, it should be noted that the measurement of the system, which is discussed in block 518, may be performed after the keys are generated for the controlling TPM. Similar to the discussion in process diagram 400, the measurements of the system may be performed after blocks 510, 524, and 528 and before blocks 512, 526, and 530. After the keys are generated and stored in the controlling TPM, the system may be measured, as shown in block 518. Then, the keys and the system measurements may be copied into the other subordinate TPM in blocks 512, 526, and 530. Accordingly, after the genkey fail bit or lock bit are cleared in blocks 516, 532 and 538, the system may boot, as shown in block 540.

[0048] While the invention may be susceptible to various modifications and alternative forms, specific embodiments have been shown by way of example in the drawings and have been described in detail herein. However, it should be understood that the invention is not intended to be limited to the particular forms disclosed. Rather, the invention is to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the invention as defined by the following appended claims.